ISO /IEC JTC 1/SC 29 /WG 1 M77026
Date: 2017-10-06

ISO/IEC JTC1/SC29/WG1
(ITU-T SG16)

Coding of Still Pictures

JBIG
Joint Bi-level Image Experts Group

JPEG
Joint Photographic Experts Group

TITLE: DMAG-UPC's answer to JPEG Privacy and Security Call for Proposals

SOURCE: Jaime Delgado, Silvia Llorente, Daniel Naro (DMAG-UPC)
Distributed Multimedia Applications Group (DMAG)
Universitat Politècnica de Catalunya – UPC BarcelonaTECH
Jordi Girona, 1-3. Mòdul D6 Campus Nord
08034 Barcelona, Spain
jaime.delgado@ac.upc.edu

PROJECT: ISO/IEC 19566-4 (JPEG Systems – Privacy, security and IPR features)

STATUS: Proposal

REQUESTED ACTION: Input contribution

DISTRIBUTION: WG1

Table of contents

1. Introduction

Document ISO/IEC JTC1 SC29/WG1 N75007 [1] issues a Call for Proposals for JPEG Privacy and Security.

This document is a partial answer to that Call. It mainly focusses on the privacy rules and protection mechanisms needed to govern the access to images. Therefore, it covers the specification of those rules and how they relate to the images in order to decide, based on the evaluation of the rules, if access is granted or not.

The concept of "privacy rules" used here refers to "restrictions to access content due to privacy conditions", as considered in Use cases 1, 2, 4, 5, 8 and 9 and Requirements 1 to 8, 14 and 15 of the Call [1].

Apart from partial (covering privacy rules and some protection aspects only), this proposal is incomplete, since the software implementation is missing. However, an example of a software architecture to be implemented is provided.

After this background introduction, the document is structured in the following parts:

- Clause 2: List of features addressed by the proposal.

- Clause 3: Detailed technical description of the proposal, including (after a presentation of the Scope):

    3.2) Privacy rules expression, including selection of a language to express the rules and mechanisms to use it in order to represent the rules needed for JPEG privacy and security.

    3.3) Linkage of privacy rules and JPEG images.

    3.4) Use of privacy rules for protection.

- Clause 4: Example of an implementation, including its software architecture.

- Clause 5: Analysis of use cases.

- Clause 6: Analysis of requirements.

- Clause 7: Summary of supported use cases and fulfilled requirements.

- Clause 8: Conclusions and possible next steps.

2. List of features addressed by the proposal

The features addressed by the proposal, according to the submission requirements defined in section 2.2 of the JPEG Privacy and Security CfP [1], are:

Protection features:

1. Solutions to support **protection tools** to **protect parts of any type of JPEG images** and/or **associated metadata** independently, while ensuring **backward and forward compatibility** with JPEG coding technologies.

2. Solutions to support handling of **hierarchical levels of access** and multiple protection levels for metadata and image protection.

Authenticity features:

4. Solutions to support **integrity checking** of image data and/or embedded metadata.

5. Solutions to support **avoiding stripping off metadata**, especially IPR information.

6. Solutions to support versioning and/or tracking changes of an image and/or associated metadata and solutions to support embedding **provenance information**.

7. Solutions to support embedding of trackable information to allow **identification and assessment of the master image** and identify derived or modified images from the master image.

3. Proposed partial solution

3.1. Scope

As stated in the Introduction section, this document only provides a partial answer to the Call [1], since it focusses on privacy rules and protection mechanisms needed to govern the access to images. Therefore, it covers the specification of those rules and how to manage them.

In particular, the focus of this proposal is on the access control aspects, with the purpose of guaranteeing image privacy. For this, there is a need to:

- Express privacy rules (the kind of protection we want and over what, to whom, and under which conditions, access is to be granted).

- Integrate the privacy rules in the management of JPEG images; or, at least, to establish an association mechanism between rules and images.

- Manage mechanisms for both metadata and content protection, including an authorization process over the rules and a systems' architecture.

The rest of subsections in this clause provide proposed specific solutions for the previous points.

3.2. Privacy rules expression

Privacy information, in the form of privacy rules, needs to be associated to the image it refers to. A simple approach is to insert this information in the file containing the image itself.

We propose two solutions to solve the problem of inserting privacy information inside a JPEG image.

In the first one, we could use the `RightsDescription` element defined in Part 2 of the JPSearch standard [2] without changing its syntax. Figure 1 shows an outline of the content and semantics of this element.

```
<RightsDescription>
  <RightsDescriptionInformation>
    Location of the rights description standard.
  </RightsDescriptionInformation>
  <Description>
    Textual description of the standard referenced.
  </Description>
  <ActualRightsDescriptionReference>
    Actual rights description in a referenced external file.
  </ActualRightsDescriptionReference>
  <ActualRightsDescription>
    Textual description of the actual rights reference.
  </ActualRightsDescription>
</RightsDescription>
```

Figure 1. JPSearch `RightsDescription` element

In this specific JPSearch core metadata element (`RightsDescription`), we have two mandatory elements: `RightsDescriptionInformation` and `ActualRightsDescriptionReference`, as shown in the formal specification replicated in Figure 2. Therefore, we could use these 2 elements for:

- 1) Identifying the standard we are using for expressing the privacy rules.

- 2) Referencing a specific external privacy rule expressed using the standard specified in 1).

Therefore, the use of the JPSearch `RightsDescription` metadata element facilitates fully independence from the chosen standard.

```
<complexType name="RightsDescriptionType">
  <sequence>
    <element name="RightsDescriptionInformation" type="anyURI"
                                          minOccurs="1" maxOccurs="1"/>
    <element name="Description" type="string" minOccurs="0"/>
    <element name="ActualRightsDescriptionReference" type="anyURI"
                                          minOccurs="1" maxOccurs="1"/>
    <element name="ActualRightsDescription" type="string" minOccurs="0"/>
  </sequence>
</complexType>
```

Figure 2. JPSearch `RightsDescriptionType` schema

The second solution to solve the problem of inserting privacy information inside a JPEG image could be to avoid JPSearch and insert the privacy rule directly in the JPEG file, for example using APP11. It should be made clear which of the 2 mechanisms is used in a specific image.

Concerning how to express the privacy rules themselves, they could be expressed in a XML-based language. We propose the use of XACML [3] for this purpose. However, any language capable of expressing privacy rules in a formal way could be used.

For illustration, an example of privacy rule and authorization request expressed in XACML is provided below.

Example of XACML privacy rule:

```
<Policy xmlns="urn:oasis:names:tc:XACML:3.0:core:schema:wd-17"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    PolicyId="urn:isdcm:policyid:1"
    RuleCombiningAlgId="urn:oasis:names:tc:XACML:1.0:rule-combining-algorithm:first-applicable"
    Version="1.0"
    xsi:schemaLocation="urn:oasis:names:tc:XACML:3.0:core:schema:wd-17
    http://docs.oasis-open.org/XACML/3.0/XACML-core-v3-schema-wd-17.xsd">

    <Description>Desert.jpg</Description>

    <Rule Effect="Permit" RuleId="urn:oasis:names:tc:XACML:2.0:ejemplo:Desert">
      <Description>
        Any user can view urn:mimage:Desert.jpg before the end of the year
      </Description>
      <Target>
        <AnyOf>
          <AllOf>
            <!-- Which resource -->
            <Match
              MatchId="urn:oasis:names:tc:XACML:1.0:function:regexp-string-match">
              <AttributeValue
                DataType="http://www.w3.org/2001/XMLSchema#string">
                  urn:mimage:Desert.jpg
              </AttributeValue>
              <AttributeDesignator
                AttributeId="urn:oasis:names:tc:XACML:1.0:resource:resource-id"
                Category="urn:oasis:names:tc:XACML:3.0:attribute-category:resource"
                DataType="http://www.w3.org/2001/XMLSchema#string"
                MustBePresent="false"/>
            </Match>

            <!-- Which action -->
            <Match MatchId="urn:oasis:names:tc:XACML:1.0:function:string-equal">
              <AttributeValue
                DataType="http://www.w3.org/2001/XMLSchema#string">
                  View
              </AttributeValue>
              <AttributeDesignator
                AttributeId="urn:oasis:names:tc:XACML:1.0:action:action-id"
                Category="urn:oasis:names:tc:XACML:3.0:attribute-category:action"
                DataType="http://www.w3.org/2001/XMLSchema#string"
                MustBePresent="false"/>
            </Match>
          </AllOf>
        </AnyOf>
      </Target>
      <Condition>
        <Apply FunctionId="urn:oasis:names:tc:XACML:1.0:function:date-less-than-or-equal">
          <Apply FunctionId="urn:oasis:names:tc:XACML:1.0:function:date-one-and-only">
            <AttributeDesignator AttributeId="accessDate"
              Category="urn:oasis:names:tc:XACML:3.0:date"
              DataType="http://www.w3.org/2001/XMLSchema#date" MustBePresent="false"/>
          </Apply>
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#date">
              2017-01-01
          </AttributeValue>
        </Apply>
      </Condition>
    </Rule>
    <Rule RuleId="urn:oasis:names:tc:XACML:2.0:FinalRule" Effect="Deny"/>
</Policy>
```

Example of XACML Request for Authorization

```
<Request xmlns="urn:oasis:names:tc:XACML:3.0:core:schema:wd-17"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" CombinedDecision="true"
   ReturnPolicyIdList="true"
   xsi:schemaLocation="urn:oasis:names:tc:XACML:3.0:core:schema:wd-17
   http://docs.oasis-open.org/XACML/3.0/XACML-core-v3-schema-wd-17.xsd">
   <Attributes Category="urn:oasis:names:tc:XACML:1.0:subject-category:access-subject">
      <Attribute AttributeId="urn:oasis:names:tc:XACML:1.0:subject:subject-id"
         IncludeInResult="false">
         <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
           Alberto
         </AttributeValue>
      </Attribute>
   </Attributes>
   <Attributes Category="urn:oasis:names:tc:XACML:3.0:date">
      <Attribute AttributeId="accessDate" IncludeInResult="false">
         <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#date">
            2016-1-14
         </AttributeValue>
      </Attribute>
   </Attributes>
   <Attributes Category="urn:oasis:names:tc:XACML:3.0:role">
      <Attribute AttributeId="role" IncludeInResult="false">
         <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            Normal User
         </AttributeValue>
      </Attribute>
   </Attributes>
   <Attributes Category="urn:oasis:names:tc:XACML:3.0:country">
      <Attribute AttributeId="country" IncludeInResult="false">
         <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            Spain
         </AttributeValue>
      </Attribute>
   </Attributes>
   <Attributes Category="urn:oasis:names:tc:XACML:3.0:attribute-category:resource">
      <Attribute AttributeId="urn:oasis:names:tc:XACML:1.0:resource:resource-id"
         IncludeInResult="false">
         <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            urn:mimage:Desert.jpg
         </AttributeValue>
      </Attribute>
   </Attributes>
   <Attributes Category="urn:oasis:names:tc:XACML:3.0:attribute-category:action">
      <Attribute AttributeId="urn:oasis:names:tc:XACML:1.0:action:action-id"
           IncludeInResult="false">
         <AttributeValue
           DataType="http://www.w3.org/2001/XMLSchema#string">
            View
         </AttributeValue>
      </Attribute>
   </Attributes>
</Request>
```

The previous example privacy rule defines the conditions to indicate that any user can **view** the image **urn:mimage:Desert.jpg** before the end of the year (**31/12/2017**).

To authorize the action, an authorization request is required. The one described above indicates that a **Normal user** wants to **view** the image **urn:mimage:Desert.jpg** from Spain on the **14/10/2017**. So, the action should be authorized, as the conditions about action and date are met.

3.3. Integration of privacy rules in the processing of JPEG images

Once we have identified where to include the privacy rules and how to formalize them, the next step is to specify how to integrate them in order to be managed.

When using the `RightsDescription` metadata element, as presented before, the rule has to be external to the image and be referenced from the metadata. Specifically, we propose to use a URL for this purpose. The URL (to be included in the `ActualRightsDescriptionReference` sub-element) would link to the rule, which could be located in a remote system, or even in the local one.

If JPSearch is not used to include a reference for an external resource, the rule itself (described with XACML, for example) could be embedded directly in the image file.

Alternatively, an image might have neither specific reference to nor embedded any privacy rule. However, this should not imply that privacy rules do not exist for that image, but that they are outside. In this case, the association to the image would be from the rule and could use an image identifier for this purpose. This approach is only useful for specific use cases, where the image is always encrypted.

3.4. Mechanisms and systems' architecture for privacy rules and protection

Privacy rules standards have associated mechanisms to evaluate the rules. In other words, applications handling protected images, once they have their rules, should perform an authorization process and, if positive, get the keys needed to decrypt the image.

If the application is in the client's side, it always needs to manage keys for the decrypting process. Key management could be more simple in a closed trusted environment or when ciphering the data with the public key of the client receiving the image.

On the contrary, if the application managing protected images is running in the server side, the steps to follow could be:

- Getting the privacy rules, from the image, from the link in the image or from outside.

- Authorizing access to image data, by running the specific authorizer software depending on the standard that the rules follow.

- Accessing the keys and decrypting the image once the user has been authorized. This step is not needed when in a trusted environment where there is no need for encryption (only controlling the access from users is enough).

Using privacy rules allows that both content and metadata could be protected. Privacy rules themselves should at least keep their integrity.

With respect to the content, the codestream could be encrypted with standard means. This proposal currently does not consider different levels of protection inside the content, although the use of privacy rules for this purpose is possible and recommended, and does not need any change to them.

On the other hand, there could be several approaches on metadata, since it might be needed to keep some metadata elements in clear in order to facilitate its use by applications, for example for search, and even for protection (privacy rules). We propose the use of signatures for integrity and authenticity. Signatures could be on independent elements and/or on the complete metadata set. If, for example, the signature in the complete metadata set is not correctly verified, metadata elements might have been stripped off. If verification of the integrity is correct, then we should verify the trustworthiness of the signer.

As already mentioned, evaluation of the privacy rules (to authorize or not access to metadata and/or content) depends on the standard used to express rules. In the case of XACML, the evaluation mechanisms are specified in the standard [3] and several software tools (even open source) exist [4].

As also already indicated, the privacy rule could be either directly embedded in a new metadata element of the image, or stored out of the image but linked from a URL inside metadata (JPSearch or not). In either case, rules authorization could be done in the client or in the server application.

Next clause presents a possible implementation of a particular server application. Two sequences of operations are considered: preparing the image and accessing the image.

4. Example implementation

In order to add privacy information to an image, at least the following software modules are required:

- User applications to interface with the image owner and the final user. Although the second application just needs rendering capabilities, both applications could be the same.

- A metadata module to manage privacy policies, including its inclusion in and request from the image.

- A privacy rules authorizer.

- A protection module, providing encryption / decryption capabilities and digital signature creation and validation. It could be used to protect both metadata and content data.

It is worth noting that the Application interfacing with the user does not need to be a desktop application at all, it could be a web interface or even a mobile app. Moreover, part of the process described in Figures 3 and 4 could be automatized.

Figure 3 shows a possible sequence diagram for applying privacy information and protecting both metadata and image content data.

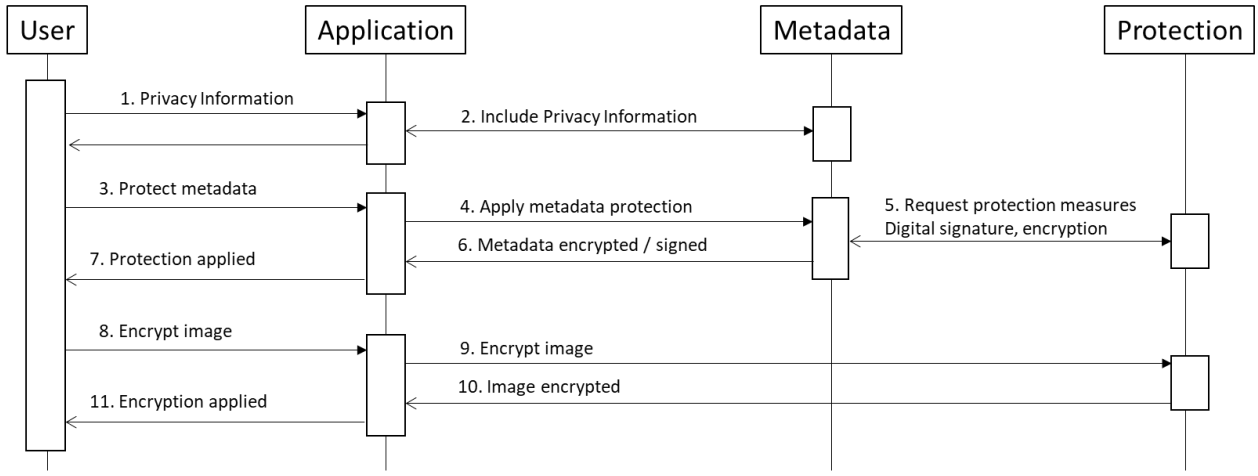

Figure 3. Image protection

The steps represented in Figure 3 are briefly described as follows:

1. The User defines some privacy information, either a XACML policy or a URL, to be inserted into an image file using an Application.
2. The Application calls the Metadata module to insert the policy into the specific metadata element. The metadata element will be created if needed.
3. The User decides to protect the metadata included in the image file. It is foreseen that several mechanisms could be applied. For instance, the XACML policy could be digitally signed and other metadata fields, like GPS positioning, could be encrypted. However, as digital signature potentially involves an overhead in size, it might be preferable to sign as many metadata fields as possible at once.
4. The Application calls the Metadata module to apply the selected protection measure(s).
5. The Metadata module requests the selected protection measure(s) to the Protection module. Depending on the measure specified, different tasks could be done, like generation of symmetric key, use of user's private key for digital signature, etc.
6. The Metadata module inserts the protected metadata into the image file and informs the application accordingly.
7. The User is informed that the protection has been applied to the specified metadata.
8. The User requests image content encryption. This is an optional step, as depending on the scenario, encryption could be not necessary.
9. The Application calls the Protection module to apply it.
10. The encrypted image content is inserted into the image file accordingly.
11. The User is notified that the encryption has been applied.

After that, the image is protected against unauthorized access, as it will be only rendered if the conditions described in the privacy policy are fulfilled.

On the other hand, Figure 4 shows how a user that wants to view the protected image has to proceed. Steps are explained next:

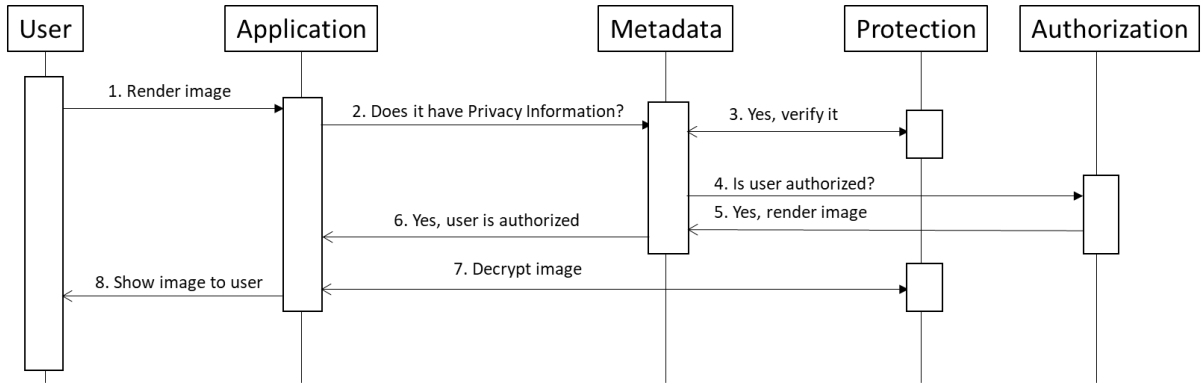

Figure 4. Authorization of access to a protected image

1. Another user wants to render the protected image.
2. The Application asks the Metadata module if it has associated privacy information.
3. The Metadata module detects privacy information, so it asks to the Protection module to verify it. A possible verification is a digital signature validation.
4. After positive verification, the Metadata module asks for authorization to the Authorization module. Authorization could be also applied to metadata elements, not only to the codestream.
5. After positive authorization, the User can view the image.

6. The Metadata module informs the Application that the User is authorized according to the privacy information contained in the image.
7. The Application requests image decryption to the Protection Module.
8. The User has been authorized to view (or the action indicated in the privacy policy) the image, so it is shown to her.

The example architecture presented may have some variations, such as:

- Rendering variations:
    - Image in local device. Authorization and decryption mechanisms should be implemented into the local application rendering the image.
    - Image in service provider. Authorization and decryption mechanisms are done in the service provider premises, and image is only rendered in case of positive authorization.
    - Image on independent repository. Combination of previous solutions may apply (remote authorization, local decryption, etc.).

- Privacy rules and protection creation variations:
    - Privacy and protection applied in local device. Rule creation and encryption mechanisms should be implemented with a local application installed in the device. The user then distributes the image to whom she wants.
    - Privacy and protection on service provider. Rule creation and encryption is done remotely according to user preferences. The protected image is stored in the service provider.

5. Analysis of use cases

Our proposal tries to handle some of the use cases presented in the Call. In this section, we analyze the ones that can be covered with the proposed solution.

Use case 1: Images repository with controlled access

We completely cover this use case.

Use case 2: Metadata protection

The solution proposed considers metadata protection, so this use case is also covered.

Use case 3: Publication and image annotation

Not covered.

Use case 4: Ephemeral photo sharing

This use case is covered when inserting privacy rules inside the image and authorizing using a specific application inside the viewer device/tool. When the rule does not apply, the image cannot be seen anymore.

Use case 5: Social networking and photo sharing

This use case is covered when the social network supports privacy rules (either embedded or external) in the image, and the authorization is provided using a specific application inside the viewer device/tool. When the rule does not apply, the image cannot be seen anymore.

Use case 6: Medical imaging

Not covered.

Use case 7: Forensic image analysis

Not covered.

Use case 8: Counter terrorism and monitoring

Partially covered. When the viewer's application requires contacting a server to render the file, it is paving the way for monitoring image's usage.

Use case 9: Privacy preserving search

Covered, image and metadata can be protected.

Use case 10: Provenance

Not covered.

Use case 11: Video surveillance

Not covered.

6. Analysis of requirements

Our proposal tries to handle some of the requirements presented in the Call. In this section, we analyze the ones we have considered.

Requirement 1: Compliance with the privacy principles defined under SC 27's privacy framework.

Covered. It provides mechanisms to protect Personally Identifiable Information like GPS positioning or identifier as described in the framework.

Requirement 2: Support JPEG-1 and JPEG 2000 backward compliant codestreams.

Covered, the proposed solution follows the structure proposed in the Call.

Requirement 3: Support encrypting metadata and codestream independently.

Covered, the proposed solution differently encrypts metadata and codestream.

Requirement 4: Support hierarchical levels of access and multiple protection levels for metadata and image protection.

Partially covered. Different rules could be defined to protect different parts of the image, although current solution only considers complete protection of the image data.

Requirement 5: Privacy policies need to be evaluated and allow access to partial or complete metadata and image data.

Partially covered. Different rules could be defined to protect different parts of the image, although current solution only considers complete protection of the image data.

Requirement 6: Support common protection tools.

Partially covered. Not all the mentioned tools are used in current proposal.

Requirement 7: Enable mechanism to support for additional protection tools.

Partially covered. The use of an external architecture should facilitate the use of additional protection tools.

Requirement 8: Support storing of non-protected information in the APP_{11} marker.

Covered. It is foreseen to protect some metadata elements like privacy rules or GPS positioning and leave other metadata elements in clear.

Requirement 9: Support box-based file format.

Not considered.

Requirement 10: Support for metadata formats.

Not considered.

Requirement 11: Support provenance functionality.

Not considered.

Requirement 12: Compatibilities to other standards (e.g. published by SC27, SC29, and W3C) and frameworks.

N/A.

Requirement 13: Support identification of the master image.

N/A.

Requirement 14: Signaling at system level and additional normative functionalities.

Covered. The proposed solution follows the JPEG formats at system level.

Requirement 15: Support to security features for metadata and content.

Partially covered, need to apply integrity mechanisms to the image.

Requirement 16: Resynchronisation points to support file carving systems.

N/A.

Requirement 17: Support for transport mechanisms.

N/A.

7. Summary of supported use cases and fulfilled requirements

This section summarizes the use case and requirements solved either partially or completely by this proposal.

Table 1. Summary of supported use cases

Use case	Covered (Yes / No / Partially)	Comments
1	Yes	Proposed architecture including privacy rules, protection and access control.
2	Yes	Metadata protection considered.
3	No	
4	Yes	Use of privacy rules, protection and access control.
5	Yes	Use of privacy rules, protection and access control.
6	No	
7	No	
8	Partially	The use of an external service to authorize privacy rules could be also used for monitoring.
9	Yes	Use of privacy rules, protection and access control.
10	No	
11	No	

Table 2. Summary of fulfilled requirements

Req.	Covered (Yes / No / Partially)	Comments
1	Yes	Proposed mechanisms to protect Personally Identifiable Information.
2	Yes	Follow the format proposed in the Call.
3	Yes	Metadata and codestream are encrypted separately.
4	Partially	Application of different rules to different parts of the image.
5	Partially	Application of different rules to different parts of the image.
6	Partially	Not all the proposed tools are used in the solution presented.
7	Partially	The use of additional protection tools could be supported with the external architecture.
8	Yes	Combine protected and unprotected metadata.

9	No	
10	No	
11	No	
12	No	
13	No	
14	Yes	The proposed solution follows the JPEG formats at system level.
15	Partially	Integrity mechanism should be applied to the image, as well as protection.
16	No	
17	No	

8. Conclusions. Next steps

This JPEG input contribution, answering to the Call on JPEG Privacy and Security, proposes to handle access control over JPEG images by adding privacy policies and (partially) encrypting metadata and/or image codestream. Also digital signatures are considered.

A few options are provided to express privacy rules and relate them to the image themselves. A client or a server application needs to perform an authorization process based on those policies and contextual information and to manage decryption keys.

We also describe a possible example implementation of the proposed solution.

This is a partial solution focusing on a few of the requirements and use cases defined in the Call.

In any case, several issues still need further work, such as details on the encryption procedures and formats of the encrypted elements, or how to integrate privacy rules with other solutions.

9. Acknowledgements

The work presented in this paper has been partially supported by the Spanish Government under the project: Secure Genomic Information Compression (GenCom, TEC2015-67774-C2-1-R).

10. References

[1] Ishikawa, T. (Editor), *ISO/IEC JTC1 SC29/WG1 N75007 JPEG Privacy and Security Call for Proposals*, March 2017.

[2] ISO/IEC, *ISO/IEC 24800-2:2011: Information technology -- JPSearch -- Part 2: Registration, identification and management of schema and ontology*, ISO/IEC, 2011.

[3] eXtensible Access Control Markup Language (XACML) v3.0, http://www.oasis-open.org/specs/index.php#xacmlv3.0, 2013.

[4] WSO2 Balana implementation, XACML authorization software, https://github.com/wso2/balana, 2017.